Big Brother is Watching You: Non-Intrusive ZigBee User Profiling

Katharina O. E. Müller[1], Delia Datsomor[1], Daria Schumm[1], Bruno Rodrigues[2], Burkhard Stiller[1]
[1]Communication Systems Group CSG, Department of Informatics IfI, University of Zurich UZH, Switzerland
[2] Embedded Sensing Group ESG, Institute of Computer Science in Vorarlberg ICV,
University of St. Gallen HSG, Switzerland
E-mail: [mueller|schumm|stiller]@ifi.uzh.ch
delia.datsomor@uzh.ch, bruno.rodrigues@unisg.ch

***Abstract*—The rise of the Internet-of-Things (IoT) and smart homes has resulted in the increased use of ZigBee as the communication protocol of choice in home networks, giving ample opportunity for network monitoring and user profiling, as a consequence, raising a major privacy concern. Yet, there has been little exploration of the extractable information solely from network packets, particularly Philips Hue packets.**

Especially as, to the authors' knowledge, there have been no studies examining whether a single network key provides enough generalization to extract data from other unknown ZigBee networks. To address this gap, this paper proposes StealthProfiler, a passive and real-time Proof-of-Concept (PoC) tool designed to identify, classify, and extract devices and events within a Philips Hue network.

As a result, the tool was successfully used to extract network events from encrypted Zigbee networks, achieving an accuracy of approximately 94% in identifying devices and events within the network without decrypting network traffic.

***Index Terms*—Internet of Things, Home Network Monitoring, Inference Rules, Data Exfiltration, User Profiling, ZigBee**

I. Introduction

In the last decade, the landscape of smart home technologies has expanded with the ongoing development of lightweight wireless communication protocols [1]. Importantly, Zigbee has become a prominent protocol for the smart home Internet of Things (IoT) devices. Devices, such as Philips Hue smart lightbulbs, rely on Zigbee for communication between bridges and individual devices, such as routers and end devices [2]. As smart home technology adoption grows, the potential for privacy violations expands beyond individual devices and may include details about user location, habits, and schedule. The overall increasing number of smart home devices each user has, make more data available for exfiltration, sniffing, eavesdropping, monitoring, and, as a consequence, more accurate user profiling.

Exploring Zigbee protocol vulnerabilities produces critical understanding of how smart home privacy can be improved. A simple exploration of the communication protocol and local network data provides enough information for user profiling. As a result, private information, such as user behavior, lighting schedules, appliance usage, and occupancy patterns, is exposed. Therefore, there is a growing need to address potential privacy vulnerabilities in smart home devices, empower users with information and awareness, and safeguard their privacy. By highlighting those vulnerabilities, the paper emphasizes the urgency for developing robust security measures to protect user privacy and the fast-changing smart home environment.

This paper investigates Zigbee communication within a smart home environment and presents StealthProfiler, a lightweight Proof-of-Concept (PoC) inference-based data analysis tool that can be easily deployed at home. The objectives include analyzing a real Philips Hue smart home environment and developing a passive and real-time tool to verify the statement that only a single network key is required to decode any other Zigbee network without requiring decryption. Thus, this paper proposes StealthProfiler, a PoC tool that leverages inference-based Zigbee profiling to recognize and track Zigbee-compatible devices, commands, and events, matching encrypted and decrypted network behaviors through passive post-mortem and real-time analysis. As a result, StealthProfiler demonstrates how the privacy of individuals is easily endangered, underscoring the potential risks in smart home environments. By exposing privacy vulnerabilities and presenting a new method for data exfiltration from commercial smart environments, the tool achieves an accuracy of almost 94% in profiling users.

Additionally, this approach provides users with insights into the usage and state of these devices within the network, facilitating the tracking of user habits and preferences while also highlighting security and privacy threats concerning data exfiltration and user profiling. Thus, the contributions of the paper are:

- Detailed analysis of link layer and network (NWK) layer Zigbee communication in the Philips Hue environment;
- Development of StealthProfiler: an inference-based passive and real-time command and event extraction tool for data exfiltration and user profiling from any ZigBee network without a network key.

The paper is organized as follows. Section II discusses the related work, highlighting the need for a new research. Section III introduces the design of StealthProfiler design, outlining core components and operations. Section IV provides results of the tool operations, its evaluation, and discussion. Finally, Section V summarizes the contribution presented and outlines future work.

978-3-903176-66-9 ©2024 IFIP

II. Background and Related Work

In recent years, the demand for smart living has driven rapid advancements in IoT devices, making them increasingly vulnerable to sniffing and eavesdropping attacks. Zigbee, a low-power wireless protocol commonly used in IoT devices, has also become a target. Zigbee communication is secured by a 128-bit network key, which encrypts messages between devices in the network, preventing unauthorized access. However, research has demonstrated methods for sniffing and analyzing Zigbee traffic. For instance, [3] presents a packet-sniffing method using software-defined radio to recognize Zigbee protocol messages, while [4] introduces an IoT forensic tool for real-time traffic capture in home automation platforms. [5] presents ZLeaks, which identifies in-home devices within encrypted Zigbee traffic by deducing application layer commands and exploiting reporting patterns and intervals. Furthermore, [6] showcases IoTSpy, a method for user activity inference through wireless context analysis, extracting packet sequence features to detect events and deduce user activities, moods, lifestyle patterns, and the presence of installed IoT devices. [7] utilizes machine learning techniques to infer user behaviors from smart home device usage, identifying and locating devices to deduce user activities. In contrast to [6] and [7], [8] focuses on beneficial applications of device monitoring. The authors propose a smart monitoring system for campus infrastructure, where the system controls the opening and closing of building doors and can integrate lighting systems and appliances.

[9] found that eavesdropping poses significant risks to user privacy. The authors demonstrate an approach where attackers could intercept the key exchange by the addition of a device to a Zigbee network, thus allowing unauthorized control and exploitation of network events. [10], utilizes machine learning in a multi-stage privacy attack and achieves over 90% accuracy in identifying the states of a device and user actions through passive network traffic sniffing. By leveraging sniffing and eavesdropping, [11] demonstrates a self-replicating worm attack in a Philips Hue network via Zigbee updates, showing how neighboring IoT devices could infect each other, spreading rapidly over large areas. [12] conducts replay attacks on Philips Hue bulbs and Xbee modules, exposing security vulnerabilities despite its built-in countermeasures. [13] analyzes the ecosystem of Philips Hue, revealing privacy risks from different control devices and smartphone applications. The authors show how various control techniques influence the amount of data transmitted to the Internet. [14] introduces ZPA, a system for privacy analysis of Zigbee-encrypted traffic in smart homes, hence addressing privacy and security issues. [15] proposes ChatterHub, a system enhancing privacy management by classifying smart-home device events through eavesdropping and machine-learning techniques.

This work was primarily inspired by ZPA [14] and Zleaks [5]. However, this work is different on multiple dimensions, as summarized in Table I. First, ZPA relies on machine learning for decryption, while Zleaks and this work utilize inference rules, which is a more lightweight approach for IoT devices. Second, Zleaks has a broad focus on Zigbee devices, including, but not limited to Philips Hue. However, this work focuses exclusively on the Philips Hue, allowing for a deeper analysis. Third, Zleaks relies on hard-coded data and frame lengths, while this approach uses JSON files to extract these lengths dynamically. Thus, providing improved testing efficiency and flexibility without affecting identification accuracy. Such an approach is favorable, as it reduces the possibility of errors when a new JSON file lacks certain package types. Additionally, saving packet type combinations with frame and data lengths in a CSV file highlights missing packet types, mitigating this concern further.

TABLE I: Comparison of the ZPA, Zleaks and StealthProfiler

	ZPA [14]	Zleaks [5]	StealthProfiler
Machine Learning	✓	×	×
Inference Rules	×	✓	✓
Periodic Reporting Patterns	×	✓	×
Dynamic System	✓	×	✓
Philips Hue	×	✓	✓
Other Manufacturers	✓	✓	×
In-Depth Analysis	×	×	✓
Broad Analysis	✓	✓	×
Network Key Extraction	✓	×	✓
Privacy Risks	✓	✓	✓
Real-time Tracking	×	×	✓

Lastly, the most important distinction between ZPA, Zleaks, and StealthProfiler is the use of a single network key to decode any other network. This has been largely unexplored in previous works and allows any Philips Hue user unhindered access to decode other ZigBee networks. StealthProfiler addresses this significant research gap.

III. StealthProfiler

StealthProfiler is a passive and real-time monitoring tool for Philips Hue network monitoring and data exfiltration. The tool focuses on identifying device types and capturing *on/off* and *level/color control* commands.

A. Initial Network Decoding

To identify devices and commands independently of the network encryption, an initial mapping between the encrypted and decrypted network communication had to be constructed. Consequently, the network key of the Zigbee network had to be retrieved. By sniffing during the addition of new light to the network, the transport and network keys were captured. Thus, allowing the decryption of all packets within that network and facilitating the mapping to the unencrypted network capture. Subsequently, the network captures could be analyzed simultaneously, enabling the mapping of network commands and events between the two.

B. Targeted Philips Hue Commands

The commands targeted by StealthProfiler encompass various Philips Hue network operations, each identified by a specific Philips Hue network identifier, including *on/off, color/level control, read attributes*, and *route record/request*. Due to the many commands available for Philips Hue devices, the *on/off* commands were selected for their prevalence, and

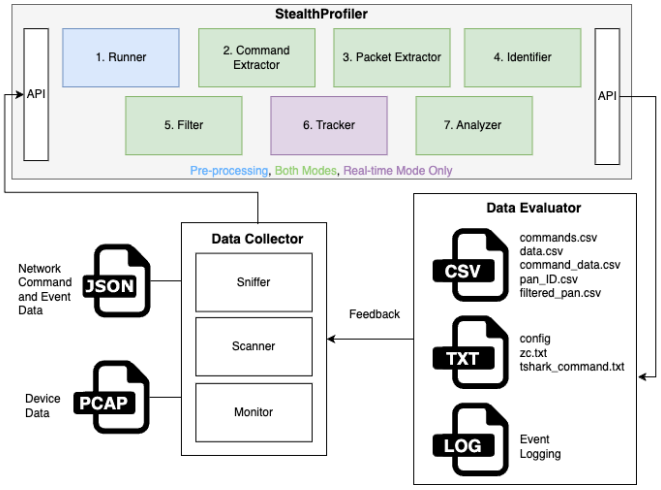


Fig. 1: StealthProfiler Core and Modes

level/color control commands were chosen as they are specific to Philips Hue networks. In addition, *broadcast* messages were thoroughly investigated to distinguish them from others.

C. Architecture

The core components included the *Packet Extractor*, *Command Extractor*, *Filter*, *Identifier*, and *Analyzer*, as depicted in Figure 1. In comparison, the *Runner* orchestrates the tool's execution and mode, and the *Tracker* component is only used during real-time mode to ensure the correct execution of the capture strategy.

1) Runner: The backbone of StealthProfiler, orchestrating the execution of all components in either real-time or passive tracking mode. In real-time mode, the *Runner* initiates *Tracker* to start a 90s network sniffing session to capture ZigBee packets. Followed by the *Command Extractor* and *Packet Extractor* to identify devices. consequently, the *Filter* is applied, and real-time tracking commences (see Section III-C6). The *Analyzer* component receives all the previous components' output and identifies network events based on a set of inference rules (see Section III-C5), which distinguish network events and commands from each other, thus inferring user behavior, such as turning *on* a light. Passive mode omits the initial 90s sniffing phase. Instead, the component analyzes the PCAP file provided by the user, following the component execution sequence of *Command Execution*, *Packet Extractor*, *Filter*, *Identifier*, and *Analyzer*.

2) Command Extractor: Its core objective is to extract crucial data from two JSON files generated by the tool from the same input network package capture. One JSON file is decrypted using the network key, while the other remains encrypted. The comparison of both JSON files allows the correlation of the decrypted commands and frame lengths with their respective data lengths, which are only accessible from the encrypted capture. The component generates three CSV files: *data.csv*, containing packet frame numbers, lengths, and data lengths, *command.csv*, storing frame numbers, lengths, and commands and *command_data.csv*, which combines both by frame number.

Firstly, *extract_command_from_packets* handles the extraction of commands and their frame lengths from the decrypted JSON file Secondly, the *extract_data_from_packets* function retrieves data lengths from the encrypted JSON file. The commands and frame lengths are then mapped to the corresponding data lengths, yielding *data_command.csv*.

The Zigbee NWK and Zigbee Cluster Library (ZCL) layers serve as primary sources for command mapping. Additionally, the component is adaptable and capable of extracting data from various JSON files. Thus, the user can easily regenerate or directly change them. Offering flexibility to cover other networks, even beyond ZigBee. If only the encrypted network is available, then only the Zigbee Network (NWK) layer is accessible. Nevertheless, the command extraction process remains the same, requiring only the frame length and data length to extract the commands.

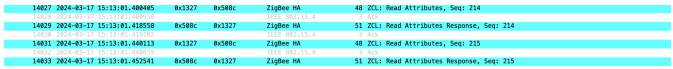
3) Packet Extractor: Streamlines the analysis of PCAP files by condensing the captured data into a CSV format. Its primary objective is to extract essential information, such as frame time and length, source, and destination, from network packets. These extracted attributes serve various analytical purposes. For instance, the frame time provides temporal context, indicating when events occurred. Meanwhile, frame length, source, destination, and data length aid in device identification and event categorization. The destination Personal Area Network (PAN) plays a crucial role by facilitating the organization of packets based on their PAN ID. This segmentation ensures that packets are grouped logically, enhancing the clarity and manageability of the data. Furthermore, the extracted frame number attribute is used for analysis, while the sequence number is used to eliminate duplicates.

Upon user input, the *Packet Extractor* determines which network should be analyzed further, forwarding the *pan_ID.csv* to the *Identifier* and *Filter* components.

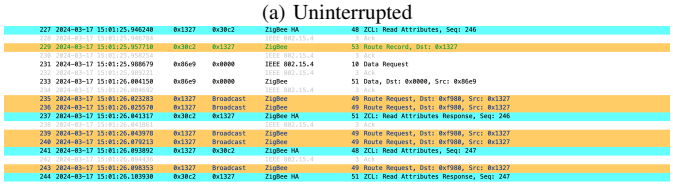
4) Identifier: The *Identifier* component receives the output *pan_ID.csv* from the *Packet Extractor* and identifies the various ZigBee devices in the given PAN. It compiles lists containing combinations of commands, frames, and data lengths, which serve as reference points for identifying packets in the subsequent *Filter* and *Analyzer* components.

The packets under consideration include *route record*, *route request*, and *read attribute response*. For example, it was observed that the *route record* and *request* packets predominantly originate from the Zigbee coordinator. Consequently, these packets serve as key indicators for identifying the coordinator within the network. Furthermore, the *read attribute response* command is commonly associated with ZigBee End Devices (ZED), which can be inferred from their inherent behavior of continuously reporting their attributes to the coordinator. This process enables the identification of devices based on their transmitted packets. The *Identifier* component then organizes the devices into lists according to their respective device types and records the coordinator's network address in a text file, *zc.txt*, which is subsequently utilized in the *Analyzer*.

5) Filter: While the *Packet Extractor* is responsible for preparing the CSV file for device identification, the main task of the *Filter* component is refining the file for event analysis. Therefore, its primary task is to process the extracted data per PAN originating from the *Packet Extractor* component and output a filtered CSV file containing only Zigbee Home Automation (*ZHA*) commands and the inference rules, allowing their identification as ZigBee events. Inference rules, refer to patterns or logical relationships that are established based on the observed behavior of network packets. These rules allow for the deduction of one event or packet type based on the occurrence of another. For example, an inference rule is formed by recognizing that a *read attributes* response packet consistently follows a *read attributes* packet in the opposite direction, seen in Figure 2. This predictable sequence can be used to infer the nature of certain packets, even when they are difficult to distinguish due to similar characteristics.



(a) Uninterrupted



(b) Interrupted

Fig. 2: Read Attribute Packet Sequence

As a result, the *Filter* component outputs *filtered_pan_ID.csv* containing only filtered *ZHA* packets, their sequence, and the commands within, enabling event detection but not device identification, as other packet types are also utilized for the latter.

6) Tracker: The *Tracker* ensures timely execution of real-time network monitoring by initiating Tshark[1] to capture 255 packets—an ideal number for analysis without significant delay. This strategy reduces the impact of background analysis, minimizes packet duplication, and enhances the clarity of the analytical process.

7) Analyzer: The *Analyzer* component recognizes the *on, off, color,* and *level control* events based on the provided *filtered_pan_ID.csv*, the *zc.txt*, and *command_and_data.csv*. It combines packet attributes, such as distinct frame and data lengths, with the respective commands. These attribute and command combinations serve as reference points for incoming packets, enabling the *Analyzer* to match a packet with the corresponding event. For example, in 1, a *broadcast* packet signaling an *on* command, corresponding with the *Lights turn on* event, is compared against incoming packets.

```
if event[0] == 'on broadcast' and dst in
    broadcast_addresses: # 47, 11
    print('User turned light on at ' + row['Time'] +
          dst)
```

Listing 1: Check *On* Packet

[1]https://www.wireshark.org/docs/man-pages/tshark.html

However, a *color control* packet, can only be distinguished from a *read attribute* packet through the application of inference rules, which check the packet sequence. If the next packet is a *read attribute response* packet, the original packet is a *read attribute* packet. If not, it is a *color control* packet.

Furthermore, it's important to consider the direction of packet flow. Specifically, event commands consistently originate from the coordinator. For example, the *color control* packet may be misconstrued as a *ZCL Groups: Get Group Membership response* packet. To distinguish the two packets, the packet source must be determined; if the source is the same as the coordinator, then it is *color control* packet and indicates a *color change* event. Upon event recognition, it is recorded along with its timestamp and corresponding network address.

IV. Results and Evaluation

This Section centers around the three main phases: setup, data collection, and data inspection, which iteratively inform each other for StealthProfiler's real-time and passive Zigbee traffic analysis. Additionally, testing outcomes will be discussed before an overall evaluation of the prototype is given. Lastly, the challenges and vulnerabilities of the Philips Hue network were explored.

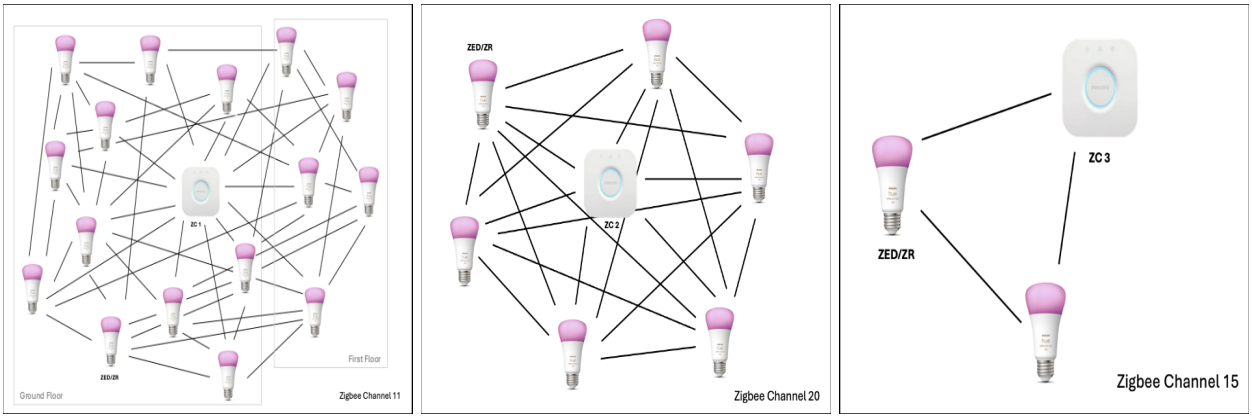
A. Data Collection

Data collection began with sniffing while executing various commands over a 15-45 minute period to determine the data gathering rate. However, a 24-hour sniffing period was considered; within 30 minutes, thousands of packets were captured, many being *read attributes* and their *responses*, which were not interesting. Tshark, Wireshark's command-line tool, can run longer and is suitable for real-time tracking but harder to analyze directly.

Sniffing started in Setup 3 to understand the Philips Hue network and test the approach, then moved to Setup 1 for a realistic smart home setting, which was more complex and unpredictable. After several iterations and feedback from data evaluation, Setup 2 was used to test StealthProfiler and its final assumptions.

B. Data Inspection and Inference Rules

Data evaluation involved analyzing packets based on assumptions to inform subsequent data collection rounds. In the first round, packets were evaluated by comparing encrypted and decrypted data to assess information visibility. Thousands of packets were analyzed alongside *Zleaks*, revealing consistent frame and data lengths across packet types, such as *unicast on* commands. Subsequently, inference rules were developed to differentiate encrypted packets. For instance, the *unicast off* command has the same frame and data length as a *read attribute* packet, only distinguishable through the packet sequence, defined and checked through inference rules. For example, a *read attributes response*, sent from an end device to the bridge, consistently follows a *read attributes* packet with opposite directional flow, establishing an inference rule.



(a) Setup 1 (b) Setup 2 (c) Setup 3

Fig. 3: Experimental Topologies

C. Testing and Outcomes

This section delves into the testing phase, examining the outcomes and evaluating the results obtained, also highlighting successes and areas for improvement.

1) Experimental Setup: Three setups were designed to test StealthProfiler and evaluate the assumption that a single network key is sufficient to decrypt any other Zigbee network. Each setup included a Philips Hue bridge, routers, and various ZEDs operating on different Zigbee channels for minimal interference and easier management.

- **Setup 1** includes one bridge, sixteen light devices, and several sensors and switches (not considered in testing). It mirrors a real-world smart home configuration, covering the first floor (living room, dining room, kitchen, and corridors) and the second floor (bedrooms). It operates on default channel 11, making it unpredictable and challenging. Figure 3 (a) shows the topology diagram.
- **Setup 2** includes one bridge, six light devices, and assorted sensors and switches, disregarded in testing. It also mirrors a real-world smart home configuration, covering the basement, and operates on channel 20.
- **Setup 3** as depicted in Figure 3 (c) determines the testing setup, featuring one bridge and two light devices. This arrangement is characterized by its clarity, predictability, and ease of manipulation. It utilizes channel 15 to facilitate easier experimentation due to its relative isolation.

During analysis, commands were issued to two devices per setup. Events from devices outside this scope were excluded. The *Identifier* was tested with a total of 27 devices. Three devices in Setup 3, 7 in Setup 2, and 17 in Setup 1. Any other devices, like sensors and switches, were excluded from consideration. Each setup was tested twice each in real-time and passive mode, with two rounds of testing per setup. Round 1 involved executing commands, while Round 2 involved monitoring the network without issuing commands to see if the prototype would identify commands that were not given. The real-time mode was tested an additional three times per setup due to its higher fragility, due to the reduced data available from sniffing for only 90 s compared to the 21 min in passive mode. Testing was conducted under optimal conditions, with restarts for failed commands to ensure all lights were reachable, though occasional sniffing disruptions occurred.

D. Device Identification Results

The detection in the ideal case of all coordinators indicates that identifying coordinators is reliable, achieving a 100% success rate. Additionally, detecting 159 out of 168 ZEDs, approximately 95%, showcases a robust performance. However, it is important to note that this success rate may be affected by the occasional unreachability of light devices during testing, which could result in the non-transmission of packets. Similarly, the detection of 174 out of 189 ZR devices, around 92%, demonstrates good results. The discrepancies are most likely caused by the sporadic unreachability of light devices during testing.

The weakest results were observed in the detection of routers. This is likely attributed to the requirement in the approach for every device to send a *link status* packet during testing for successful detection. Factors such as intermittent connection loss or device range limitations may render routers temporarily unreachable, impacting detection rates. Although routers show clear conditions to identify them, they received the weakest results.

Comparing the real-time and passive detection methods, the real-time mode achieved a detection rate of 249 out of 270, approximately 92%, while the passive mode detected 105 out of 108 devices, around 97%. Although the difference is minimal, passive detection outperformed real-time detection, possibly due to the longer duration of sniffing, allowing for a more significant number of packets to be captured and more data to be analyzed. Consequently, conducting more extensive testing in real-time mode is advisable to refine device identification further. A short overview of the device detection outcomes is presented in Table II.

TABLE II: Device Detection Outcomes

Round	real-time/Passive	Coordinators	ZEDs	Routers
1	real-time	3	23	26
2	real-time	3	22	25
3	real-time	3	21	21
4	real-time	3	23	25
5	real-time	3	22	26
1	Passive	3	24	25
2	Passive	3	24	26

E. Event Detection Results

The event detection results show promise, with a detection rate of 97% in Round 1, where 35 out of 36 events were detected, and a perfect non-detection rate in Round 2, with 36 out of 36 non-events not detected. This leads to an overall accuracy of 98% for both detection and non-detection (see Table III). A few undefined occurrences were observed, particularly in Setup 1, the largest setup and likely to have more complex interactions. These undefined occurrences could be false identifications or caused by household activities.

TABLE III: Event Detection Outcomes

Real-time/Passive	Round	Correctly detected events/non-events
real-time	1	17
real-time	2	18
Passive	1	18
Passive	2	18

Regarding the comparison between real-time and passive modes, there was minimal difference observed. One event in real-time mode was not detected, while all events were detected in passive mode. This missed event may be attributed to the brief pause after capturing 255 packets in real-time mode, during which the system writes the file and resumes capturing. Commands issued during these milliseconds could potentially be overlooked, presenting quite a big limitation.

Passive mode detected more events that shouldn't have been detected, particularly *color control* events. This may be linked to the identification of packets with similar frames, data lengths, and directions originating from the bridge. Despite efforts to differentiate these packets, no definitive rules were found to entirely distinguish *color control* packets from all other packet types with the same frame and data length combination, likely due to skipped packets. Nevertheless, since the results were not overwhelmed with skipped packets, tolerating these discrepancies is deemed acceptable.

F. Analysis

StealthProfiler was analyzed with respect to precision, recall, and accuracy evaluation metrics. The testing resulted in 389 True Positives (TP), 3 False Positives (FP), 25 False Negatives (FN), and 36 True Negatives.

1) Evaluation Metrics: **Recall** focuses on the proportion of true positive predictions among all actual positive instances. The recall value calculated is 94%, indicating a high proportion of relevant cases being identified.

$$\frac{TP}{TP+FN} = \frac{389}{389+25} \approx 0.940$$

Precision focuses on not labeling a negative sample as positive. The precision value is around 99%, suggesting that StealthProfiler has high precision in correctly identifying positive samples.

$$\frac{TP}{TP+FP} = \frac{389}{389+3} \approx 0.992$$

Accuracy provides an overall assessment of correctness, considering both true positive and true negative predictions. The accuracy value is rounded to 94%, indicating a high level of correctness in the predictions.

$$\frac{TP+TN}{TP+FP+FN+TN} = \frac{389+36}{389+3+25+36} \approx 0.938$$

2) Evaluation Discussion: These metrics demonstrate good real-time and passive tracking performance, contingent on conditions such as reachable lights, functional Philips Hue commands, and sufficient packet capture. StealthProfiler, while designed with a holistic and iterative approach, is limited by these dependencies. Similar challenges will affect machine learning approaches, if conditions are not met. Interdependence among components and their sequential interactions pose another limitation. For instance, *Identifier* must store the coordinator in a text file for the *Analyzer* to function correctly. Omitting this step compromises the *Analyzer's* performance. However, this interdependence also allows components to be executed individually, provided all required input files are present. Access to filtered PAN packets and Zigbee coordinator information from previous runs enables independent execution of the *Analyzer*, aiding interactive improvement.

Overall, StealthProfiler meets its primary objectives and shows significant promise. Its iterative methodology and structure enhance transparency, making it accessible to newcomers and contributors. Overall, StealthProfiler demonstrates good performance during testing.

G. Emerging Security and Privacy Concerns

The development and testing have demonstrated the feasibility and implications of real-time and passive tracking within a Philips Hue smart home. This underscores the importance of understanding the relationship between network architecture and data transmission, especially in IoT devices.

Passive and active tracking is feasible. StealthProfiler showed that Philips Hue's smart lighting systems can be tracked both passively and in real-time by analyzing network packets and formulating rules based on their sequences. Using an nRF board, Wireshark, and a network key, exposes several security and privacy vulnerabilities in the Philips Hue ecosystem. The ability to track these systems highlights profound ethical and privacy concerns in IoT devices.

One key concern is that possessing a single network key allows decryption of other Zigbee Philips Hue networks, regardless of their network keys. Based on consistent frame and data lengths for various packet types, any Philips Hue user could potentially decrypt another user's network.

Existing vulnerabilities in Zigbee The analysis also highlights other network vulnerabilities within the smart home

environment. Gaining access to the Philips Hue network key, a known vulnerability, remains easily exploited. StealthProfiler's ability to analyze Zigbee packets introduces the risk of data interception, potentially compromising sensitive information, such as device network addresses and user commands. The collected sensitive information and transmitted data could then be shared with third parties without their explicit consent, which could lead to unauthorized access or misuse of users' personal information by third-party entities, raising concerns about data privacy and security.

Exploiting these vulnerabilities could grant malicious actors control over the smart lighting system. Examining a decrypted network JSON files can facilitate replay attacks and unauthorized network access. Hence, StealthProfiler could be used for harmful actions, such as denial-of-service attacks, worms, or disrupting other network devices, compromising the smart home ecosystem's stability and security.

By analyzing events like *on/off*, *color changes*, and *brightness adjustments*, StealthProfiler inadvertently tracks users' activities, revealing habits, routines, and occupancy patterns, thus violating their privacy. This can lead to behavioral profiling, revealing sensitive information about users' daily lives, such as presence at home, sleep patterns, and moods. If malicious actors gain access to this information, it could facilitate targeted break-ins, compromising physical security.

While StealthProfiler shows positive outcomes, it also reveals notable security and privacy vulnerabilities. Beyond technical risks, it highlights the extent of information that can be extracted from packet traffic in a smart home environment, emphasizing the need for robust security measures.

V. Conclusion and Future Work

StealthProfiler offers a PoC passive and real-time tool for identifying, classifying, and extracting devices and events from encrypted Zigbee networks, particularly focusing on the Philips Hue smart lightbulbs. With StealthProfiler, this paper validates the statement that possessing a single network key can allow the decoding of any Philips Hue network and identify a device type and various events within the network.

Therefore, uncovering significant security and privacy risks, such as data interception, exposing sensitive data, and potential exploitation through use profiling. Additionally, it forms the basis for long-term habit tracking, by expanding the command recognition for long-term aggregation and pattern recognition, in future extensions of the work. Moreover, the integration of additional devices from other vendors and additional messaging protocols can improve interoperability and inter-protocol pattern recognition to gain further insights into communication protocol security.

Acknowledgements

This paper was supported partially by *(a)* the University of Zürich UZH, Switzerland, and *(b)* the Swiss State Secretariat for Education, Research and Innovation (SERI) under grant agreement number 22.00165 within the context of the EU CERTIFY project.

References

[1] A. A. Zaidan, B. B. Zaidan, M. Y. Qahtan, O. S. Albahri, A. S. Albahri, M. Alaa, F. M. Jumaah, M. Talal, K. L. Tan, W. L. Shir, and C. K. Lim, "A survey on communication components for IoT-based technologies in smart homes," *Telecommunication Systems*, vol. 69, no. 1, pp. 1–25, Sep. 2018. [Online]. Available: https://doi.org/10.1007/s11235-018-0430-8

[2] "Zigbee 3.0 support in Hue ecosystem," [Accessed: 2024-04-07]. [Online]. Available: https://developers.meethue.com/zigbee-3-0-support-in-hue-ecosystem/

[3] K. Cheng, Y. Deng, L. Zhang, X. Cui, J. Chen, and W. Luo, "Research on ZigBee Device Recognition Based on Software Defined Radio," *Journal of Physics: Conference Series*, vol. 2290, no. 1, p. 012040, Jun. 2022, publisher: IOP Publishing. [Online]. Available: https://dx.doi.org/10.1088/1742-6596/2290/1/012040

[4] A. Boiano, A. E. C. Redondi, and M. Cesana, "IoTScent: Enhancing Forensic Capabilities in Internet of Things Gateways," Oct. 2023, arXiv:2310.03401 [cs]. [Online]. Available: http://arxiv.org/abs/2310.03401

[5] N. Shafqat, D. J. Dubois, D. Choffnes, A. Schulman, D. Bharadia, and A. Ranganathan, "ZLeaks: Passive Inference Attacks on Zigbee Based Smart Homes," in *Applied Cryptography and Network Security*, ser. Lecture Notes in Computer Science, G. Ateniese and D. Venturi, Eds. Cham: Springer International Publishing, 2022, pp. 105–125.

[6] T. Gu, Z. Fang, A. Abhishek, and P. Mohapatra, "IoTSpy: Uncovering Human Privacy Leakage in IoT Networks via Mining Wireless Context," in *2020 IEEE 31st Annual International Symposium on Personal, Indoor and Mobile Radio Communications*, Aug. 2020, pp. 1–7, iSSN: 2166-9589.

[7] X. Guo, J. Quan, J. Hou, H. Zhou, X. He, and T. He, "Accurately Identify and Localize Commodity Devices from Encrypted Smart Home Traffic," in *2022 18th International Conference on Mobility, Sensing and Networking (MSN)*, Dec. 2022, pp. 663–670. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10076752

[8] A. A. Allahham and M. A. Rahman, "A smart monitoring system for campus using ZigBee wireless sensor networks," *International Journal of Software Engineering and Computer Systems*, vol. 4, pp. 1–14, 02 2018. [Online]. Available: https://core.ac.uk/reader/159195000

[9] L. J. A. Jansen, "Assessing smart home security : a Zigbee case study," Jan. 2022, publisher: University of Twente. [Online]. Available: https://essay.utwente.nl/89274/

[10] A. Acar, H. Fereidooni, T. Abera, A. K. Sikder, M. Miettinen, H. Aksu, M. Conti, A.-R. Sadeghi, and S. Uluagac, "Peek-a-boo: i see your smart home activities, even encrypted!" in *Proceedings of the 13th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, ser. WiSec '20. New York, NY, USA: Association for Computing Machinery, Jul. 2020, pp. 207–218. [Online]. Available: https://dl.acm.org/doi/10.1145/3395351.3399421

[11] E. Ronen, A. Shamir, A.-O. Weingarten, and C. OFlynn, "IoT Goes Nuclear: Creating a ZigBee Chain Reaction," in *2017 IEEE Symposium on Security and Privacy (SP)*. San Jose, CA, USA: IEEE, May 2017, pp. 195–212. [Online]. Available: http://ieeexplore.ieee.org/document/7958578/

[12] M. S. Wara and Q. Yu, "New Replay Attacks on ZigBee Devices for Internet-of-Things (IoT) Applications," in *2020 IEEE International Conference on Embedded Software and Systems (ICESS)*. Shanghai, China: IEEE, Dec. 2020, pp. 1–6. [Online]. Available: https://ieeexplore.ieee.org/document/9301593/

[13] M. Thiery, V. Roca, and A. Legout, "Privacy implications of switching ON a light bulb in the IoT world." [Online]. Available: https://inria.hal.science/hal-02196544

[14] R. Li, W. Zhang, L. Wu, Y. Tang, and X. Xie, "ZPA: A Smart Home Privacy Analysis System Based on ZigBee Encrypted Traffic," *Wireless Communications and Mobile Computing*, vol. 2023, p. e6731783, Jan. 2023, publisher: Hindawi. [Online]. Available: https://www.hindawi.com/journals/wcmc/2023/6731783/

[15] O. Setayeshfar, K. Subramani, X. Yuan, R. Dey, D. Hong, I. K. Kim, and K. H. Lee, "Privacy invasion via smart-home hub in personal area networks," *Pervasive and Mobile Computing*, vol. 85, p. 101675, Sep. 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1574119222000955